

Normal forms for linear displacement context-free grammars

Alexey Sorokin

July 31, 2015

Abstract

In this paper we prove several results on normal forms for linear displacement context-free grammars. The results themselves are rather simple and use well-known techniques, but they are extensively used in more complex constructions. Therefore this article mostly serves educational and referential purposes.

1 Displacement context-free grammars

Displacement context-free grammars (DCFGs) are a reformulation of well-nested multiple context-free grammars. In this draft we use tuple notation. Let Σ be a finite alphabet, then Σ^* denotes the set of all words with letters in Σ , ε being the empty string. When Σ is fixed, Θ_k denotes the set of all tuples of the form (u_0, \dots, u_k) , $u_i \in \Sigma^*$ and $\Theta = \bigcup_{k \in \mathbb{N}} \Theta_k$. We call k the rank of the tuple $u = (u_0, \dots, u_k)$ and denote it by $\text{rk}(u)$. The length $|u|$ of a tuple u is the sum of lengths of all its components, we denote by $\Theta^{(l)}$ the set of all tuples of length l . The notation $\Theta^{(\leq l)}$ and $\Theta^{(\geq l)}$ are also understood in a natural way.

We use the displacement context-free languages notation for well-nested MCFLs. We consider tuples of strings instead of gapped strings. Let Σ be a finite alphabet, then Σ^* denotes the set of all words with letters in Σ , ε being the empty string. When Σ is fixed, Θ_k denotes the set of all tuples of the form (u_0, \dots, u_k) , $u_i \in \Sigma^*$ and $\Theta = \bigcup_{k \in \mathbb{N}} \Theta_k$. We call k the rank of the tuple $u = (u_0, \dots, u_k)$ and denote it by $\text{rk}(u)$. The length $|u|$ of a tuple u is the sum of lengths of all its components, we denote by $\Theta^{(l)}$ the set of all tuples of length l and also write $\Theta^{(\leq l)}$ for $\bigcup_{j \leq l} \Theta^{(j)}$.

On the set of tuples we define the concatenation operation $\cdot: \Theta_i \times \Theta_j \rightarrow \Theta_{i+j}$ and the countable set of intercalation operations $\odot_l: \Theta_i \times \Theta_j \rightarrow \Theta_{i+j-1}$:

$$\begin{aligned} (x_0, \dots, x_i) \cdot (y_0, \dots, y_j) &= (x_0, \dots, x_i y_0, \dots, y_j) \\ (x_0, \dots, x_i) \odot_l (y_0, \dots, y_j) &= (x_0, \dots, x_{l-1} y_0, y_1, \dots, y_j x_l, \dots, x_i) \end{aligned}$$

Let N be a finite ranked set of nonterminals and $\text{rk}: N \rightarrow \mathbb{N}$ be the rank function. Let $Op_k = \{\cdot, \odot_1, \dots, \odot_k\}$, the set $Tm_k(N, \Sigma)$ of k -correct terms is defined as follows:

1. $\forall j \leq k \ (\Theta_j \subset Tm_k(N, \Sigma))$.
2. If $\alpha, \beta \in Tm_k$ and $\text{rk}(\alpha) + \text{rk}(\beta) \leq k$, then $(\alpha \cdot \beta) \in Tm_k$, $\text{rk}(\alpha \cdot \beta) = \text{rk}(\alpha) + \text{rk}(\beta)$.
3. If $j \leq k$, $\alpha, \beta \in Tm_k$, $\text{rk}(\alpha) + \text{rk}(\beta) \leq k + 1$, $\text{rk}(\alpha) \geq j$, then $(\alpha \odot_j \beta) \in Tm_k$, $\text{rk}(\alpha \odot_j \beta) = \text{rk}(\alpha) + \text{rk}(\beta) - 1$.

We assume that all the operation symbols are leftassociative and concatenation has greater priority than intercalation. We may also omit the \cdot symbol, so the notation $A \odot_2 BC \odot_1 D$ means $(A \odot_2 ((B \cdot C)) \odot_1 D)$.

Let $\text{Var} = \{x_1, x_2, \dots\}$ be a countable ranked set of variables, such that for every k there is an infinite number of variables having rank k . A context $C[x]$ is a term where a variable x occurs in a leaf position, the rank of x must respect the constraints of term construction. Provided $\beta \in Tm_k$ and $\text{rk}(x) = \text{rk}(\beta)$, $C[\beta]$ denotes the

result of substituting β for x in C . A valuation function ν assigns words of rank l to the variables of rank l for any $l \leq k$ in an arbitrary way. It also maps all the elements of Θ to themselves. Interpreting the connectives from Op_k as corresponding binary operations, we are able to calculate the valuation of every ground term (i. e. containing no nonterminal occurrences). It is easy to prove that $\text{rk}(\alpha) = \text{rk}(\nu(\alpha))$ holds for every α . The set of k -correct ground terms is denoted by $\text{GrTm}_k(\Sigma)$.

Definition. A k -displacement context-free grammar (k -DCFG) is a quadruple $G = \langle N, \Sigma, P, S \rangle$, where Σ is a finite alphabet, N is a finite ranked set of nonterminals and $\Sigma \cap N = \emptyset$, $S \in N$ is a start symbol such that $\text{rk}(S) = 0$ and P is a set of rules of the form $A \rightarrow \alpha$. Here A is a nonterminal, α is a term from $\text{Tm}_k(N, \Sigma)$, such that $\text{rk}(A) = \text{rk}(\alpha)$.

Definition. The derivability relation $\vdash_G \in N \times \text{Tm}_k$ associated with the grammar G is the smallest reflexive transitive relation such that the facts $(B \rightarrow \beta) \in P$ and $A \vdash C[B]$ imply that $A \vdash C[\beta]$ for any context C . Let $L_G(A) = \{\nu(\alpha) \mid A \vdash_G \alpha, \alpha \in \text{GrTm}_k\}$ denote the set of word, which are derivable from a nonterminal A , then $L(G) = L_G(S)$.

Example. A k -DCFG $G_k = \langle \{S, T\}, \{a_i, b_i \mid i \in [0; k]\}, P, S \rangle$, where the set P is defined below, derives the language $L_k = \{a_0^m b_0^m \dots a_k^m b_k^m\}$.

$$\begin{aligned} S &\rightarrow \underbrace{(\dots (T \odot_1 \varepsilon) \dots)}_{(k-1) \text{ times}} \odot_1 \varepsilon \\ T &\rightarrow a_0(T \odot_1 (b_0, a_1) \dots \odot_k (b_{k-1}, a_k))b_k \\ T &\rightarrow \underbrace{(\varepsilon, \dots, \varepsilon)}_{(k+1) \text{ times}} \end{aligned}$$

In what follows we assume that all the string tuples which occur in term leaves belong to $\Theta^{(\leq 1)}$. Obviously, this constraint does not restrict the generative power of DCFGs.

Definition. A term is called linear if it contains zero or one occurrences of nonterminals. A grammar is linear if right sides of all its rules are linear terms.

In this paper we study normal forms for linear DCFGs. The following result for DCFGs in general was obtained in [1].

Theorem 1. *Every k -DCFG is equivalent to some k -DCFG $G = \langle N, \Sigma, P, S \rangle$ which has the rules only of the following form:*

1. $A \rightarrow B \cdot C$, where $A \in N - \{X\}$, $B, C \in N - \{S\}$,
2. $A \rightarrow B \odot_j C$, where $j \leq k$, $A \in N - \{X\}$, $B, C \in N - \{S, X\}$,
3. $A \rightarrow a$, where $a \in \Sigma$,
4. $X \rightarrow (\varepsilon, \varepsilon)$,
5. $S \rightarrow \varepsilon$.

2 Normal forms for linear DCFGs

A valuation may be extended to variables and nonterminals by assigning every variable an arbitrary word of appropriate rank. When the valuation is fixed, the value of a context is calculated just like the term value. Two contexts are equivalent if they have the same value under all valuations. Obviously, if we replace the right-hand term in a grammar rule by an equivalent term, the generated language does not change. Basic equivalencies are listed below:

Statement 1. *The following ground multicontexts are equivalent:*

1. $(x_1 \cdot x_2) \cdot x_3 \sim x_1 \cdot (x_2 \cdot x_3),$
2. $(x_1 \cdot x_2) \odot_j x_3 \sim (x_1 \odot_j x_3) \cdot x_2$ if $j \leq \text{rk}(x_1),$
3. $(x_1 \cdot x_2) \odot_j x_3 \sim x_1 \cdot (x_2 \odot_{j-\text{rk}(x_1)} x_3)$ if $\text{rk}(x_1) < j \leq \text{rk}(x_1) + \text{rk}(x_2),$
4. $(x_1 \odot_l x_2) \odot_j x_3 \sim (x_1 \odot_j x_3) \odot_{l+\text{rk}(x_3)-1} x_2$ if $j < l,$
5. $(x_1 \odot_l x_2) \odot_j x_3 \sim x_1 \odot_l (x_2 \odot_{j-l+1} x_3)$ if $l \leq j < l + \text{rk}(x_2),$
6. $(x_1 \odot_l x_2) \odot_j x_3 \sim (x_1 \odot_{j-\text{rk}(x_2)+1} x_3) \odot_l x_2$ if $j \geq l + \text{rk}(x_2).$
7. $(\varepsilon, \varepsilon) \odot_1 x_1 \sim x_1,$
8. $x_1 \odot_j (\varepsilon, \varepsilon) \sim x_1$ for any $j \leq \text{rk}(x_1).$

Lemma 1. *Every linear k -DCFG is equivalent to some k -DCFG with the rules only of the form*

- $A \rightarrow uB$ or $A \rightarrow Bu, |u| \leq 1, u \neq \varepsilon,$
- $A \rightarrow B \odot_j u, |u| \leq 1,$
- $A \rightarrow u, |u| \leq 1,$

Proof. Through the proof we define a well-formed term by the following definition:

- A nonterminal or an element of $\Theta^{(\leq 1)}$ is well-formed,
- If α is a well-formed term, then any k -correct term of the form $u\alpha$ or αu , where $u \in \Theta^{(\leq 1)}$ and $u \neq \varepsilon$, is well-formed,
- If α is a well-formed term, then any k -correct term of the form $\alpha \odot_j u$, where $u \in \Theta^{(\leq 1)}$, is well-formed,

It is sufficient to prove that every linear term α is equivalent to some well-formed term. This is done by induction on term construction using the basic equivalencies and the fact that $(u_0, \dots, u_l) \odot_j \alpha \sim (u_0, \dots, u_{j-1}) \alpha (u_j, \dots, u_l)$ for any term α . \square

In what follows we sometimes denote the tuple $(\varepsilon, \varepsilon)$ by 1.

Lemma 2. *Every linear k -DCFG G is equivalent to some k -DCFG with the rules only of the form*

- $A \rightarrow uB$ or $A \rightarrow Bu, |u| \leq 1, u \neq \varepsilon,$
- $A \rightarrow B \odot_j u, |u| \leq 1,$
- $A \rightarrow u, |u| = 1,$
- $S \rightarrow \varepsilon.$

Proof. The proof is analogous to ε -rules elimination in standard DCFGs. We assume that G already has the form introduced in the previous lemma. We want to create a new grammar with the set of rules P' where every nonterminal $A \neq S$ of rank l generates all the tuples except $\underbrace{(\varepsilon, \dots, \varepsilon)}_{l+1 \text{ times}} = 1^l$. At first we determine for every

nonterminal, whether it generates the word 1^l , such nonterminals are called ε -generating. If A generates only this word, then it is called strictly ε -generating.

We process every element of the old set of rules P by the following algorithm.

1. If the rule has the form $A \rightarrow B_l * u$, $* \in Op_k$ and B is not strictly ε -generating, then this rule is added to P' .
2. If the rule has the form $A \rightarrow B_l u$, $|u| = 1$ and B is ε -generating, then we also add the rules, obtained by binarizing the rule $A \rightarrow (\varepsilon, \varepsilon)^p u$.
3. If the rule has the form $A \rightarrow u B_l$, $|u| = 1$ and B is ε -generating, then we also add the rule $A \rightarrow u(\varepsilon, \varepsilon)^p$.
4. If the rule has the form $A \rightarrow B_l \odot_j u$, $|u| = 1$ and B is ε -generating, then we also add the rule $A \rightarrow (\varepsilon, \varepsilon)^{j-1} u(\varepsilon, \varepsilon)^{l-j}$.
5. We include to P' all the rules in P of the form $A \rightarrow u$, $|u| = 1$.
6. We also include the rule $S \rightarrow \varepsilon$, if $\varepsilon \in L(G)$.

The correctness of the constructed grammar is proved by standard induction on word length. □

Lemma 3. *Every linear k -DCFG G is equivalent to some k -DCFG with the rules only of the form*

- $A \rightarrow uB$ or $A \rightarrow Bu$, $|u| \leq 1$, $u \neq \varepsilon$,
- $A \rightarrow B \odot_j u$, $|u| = 1$,
- $A \rightarrow u$, $|u| = 1$,
- $S \rightarrow \varepsilon$.

Proof. We assume that G already satisfies Lemma 2. At first we want to remove the rules of the form $A \rightarrow B \odot_j \varepsilon$. To reach this goal we create for every nonterminal B and every $j \in [1; \text{rk}(B)]$ its j -th bridge \hat{B}^j with the following properties: if B generates the word $(u_0, \dots, u_{j-1}, u_j, \dots, u_l)$, then \hat{B}^j generates the word $(u_0, \dots, u_{j-1}u_j, \dots, u_l)$ and vice versa. Then we create bridges for the newly introduced nonterminals and so on. Since the bridged nonterminal has lower rank than the initial one, this process will terminate.

To satisfy the declared properties we extend the grammar with the following rules. The notation \hat{u}^j denotes the word obtained from $u = (u_0, \dots, u_{j-1}, u_j, \dots, u_l)$ by removing the j -th gap. The subscript here and to the end of the paper marks the rank of the nonterminal.

1. For every rule $A \rightarrow uB$ we add the rule $\hat{A}^j \rightarrow \hat{u}^j B$ in case $j \leq \text{rk}(u)$ and the rule $\hat{A}^j \rightarrow u\hat{B}^{l-j}$ in case $l \geq \text{rk}(u)$.
2. For every rule $A \rightarrow B_r u$ we add the rule $\hat{A}^j \rightarrow B\hat{u}^{j-r}$ in case $j > r$ and the rule $\hat{A}^j \rightarrow \hat{B}^j u$ in case $j \leq r$.
3. For every rule $A \rightarrow B \odot_l u$ we add the rule $\hat{A}^j \rightarrow \hat{B}^j \odot_{l-1} u$ in case $j < m$, the rule $\hat{A}^j \rightarrow B \odot_l \hat{u}^{j-l+1}$ in case $l \leq j < l + \text{rk}(u)$ and $\hat{A}^j \rightarrow \hat{B}^{j-\text{rk}(u)+1} \odot_l u$ in case $j \geq l + \text{rk}(u)$.
4. For every rule $A \rightarrow u$ we add the rule $\hat{A}^j \rightarrow \hat{u}^j$.
5. If the grammar contained the rule $S \rightarrow \varepsilon$, we preserve this rule.

Afterwards we remove replace every rule of the form $A \rightarrow B \odot_j \varepsilon$ with the rule $A \rightarrow \hat{B}^j$. We also replace all the rules of the form $A \rightarrow B \odot_j (\varepsilon, \varepsilon)$ by the rule $A \rightarrow B$ and then eliminate unary rules by standard procedure.

It remains to remove the rules of the form $A \rightarrow B \odot_j 1^l$ for $l \geq 2$. It is done analogously to the previous step. On the set of tuples we define the j, l -split operation $\bar{u}^{j,l}$, which inserts the tuple 1^l into the j -th gap of u , this operation is naturally extended to languages. For every nonterminal B we introduce its j, l -split $\bar{B}^{j,l}$ (in case $\text{rk}(B) + l \leq k + 1$) which generates the (j, l) -split of $L(B)$. We repeat this procedure until all nonterminals of rank less than K have splitted versions. It is done just in the same way we have introduced the bridge nonterminals.

Now we replace every rule of the form $A \rightarrow B \odot_j 1^l$ by the rule $A \rightarrow \bar{B}^{j,l}$ and eliminate unary rules as earlier. The lemma is proved. □

Finally, we want to eliminate tuples of length 0 at all. For every natural p we introduce an unary operation $_{/p}$, which transforms a tuple of the form $u = va1^p$ to the string $u_{/p} = va$ in case $a \in \Sigma$, otherwise this operation is undefined. Informally, it deletes p rightmost ε components of the tuple provided the rightmost fragment of the obtained tuple will be nonempty. The operation \backslash_p is defined symmetrically. Both the operations are naturally extended from individual tuples to languages.

Theorem 2. *Every linear k -DCFG G is equivalent to some k -DCFG with the rules only of the form*

- $A \rightarrow uB$ or $A \rightarrow Bu$, $|u| = 1$,
- $A \rightarrow B \odot_j u$, $|u| = 1$,
- $A \rightarrow u$, $|u| = 1$,
- $S \rightarrow \varepsilon$.

Proof. We assume that initial grammar $G = \langle N, \Sigma, P, S \rangle$ already satisfies Lemma 3. We set $N' = \{A_{/p} \mid A \in N, p \leq \text{rk}(A)\}$, $S' = S_{/0}$ and construct the set P' by the following procedure:

1. For every rule of the form $A \rightarrow uB$ we add the rule $A_{/p} \rightarrow uB_{/p}$ for all possible p .
2. For every rule of the form $A \rightarrow B(1^q a 1^p)$ (every rule of the form $A \rightarrow Bu$ with $|u| = 1$ can be expressed so) we add the rule $A_{/p} \rightarrow B(1^q a)$.
3. For every rule of the form $A \rightarrow B1^q$ and every $p \geq q$, we add the rule $A_{/p} \rightarrow B_{/(p-q)}$.
4. For every rule of the form $A \rightarrow B \odot_j u$ and every $p < \text{rk}(B) - j$ we add the rule $A_{/p} \rightarrow B_{/p} \odot_j u$.
5. For every rule of the form $A \rightarrow B \odot_j (1^q a 1^r)$ we add the rule $A_{/p} \rightarrow B \odot_j (1^q a)$ with $p = r + (\text{rk}(B) - j)$.
6. For every rule of the form $A \rightarrow a$ we add the rule $A_{/0} \rightarrow a$.
7. If $(S \rightarrow \varepsilon) \in P$, then we also add the rule $S_{/0} \rightarrow \varepsilon$.

It is straightforward to check that $L(A_{/p}) = (L(A))_{/p}$, hence $L(S_{/0}) = (L(S))_{/0} = L(S)$ as required. We have eliminated rules of the form $A \rightarrow B1^p$, the rules of the form $A \rightarrow 1^p B$ are removed analogously. The theorem is proved. \square

References

- [1] Alexey Sorokin. Normal forms for multiple context-free languages and displacement Lambek grammars. In Sergei Artemov and Anil Nerode, editors, *Logical Foundations of Computer Science*, volume 7734 of *Lecture Notes in Computer Science*, pages 319–334. Springer Berlin Heidelberg, 2013.